Claims

1. A method for storing keys for authentication or encryption, the method being carried out by a host system, and comprising the steps of:-

the host system operating as a key server controlling storage of the keys in a software database file system.

2. A method as claimed in claim 1, wherein the key server manages separate and individual security for each key with per-key encryption.

3. A method as claimed in claim 1, wherein the key server associates a set of keys with an alias, and each alias has an associated pass phrase.

4. A method as claimed in claim 1, wherein a request to create a key is made by an alias, the server causes a key to be generated by a cryptographic accelerator, and stores the key in the database.

5. A method as claimed in claim 1, wherein the key server signs and hashes all files, and then hashes them to signed and encrypted files.

6. A method as claimed in claim 3, wherein aliases identify key rings which hold keys and certificates associated with the alias.

7. A method as claimed in claim 6, wherein each key ring is an indexed structure.

8. A method as claimed in claim 6, wherein each key ring allows access to certificate descriptions which refer to files and contain information on inception, dates, expiry dates, and creation dates.

9. A method as claimed in claim 1, wherein the key server, upon deletion of a key, spawns a thread which writes zeros or random numbers into a file which contains the key to overwrite the key.

10. A method as claimed in claim 9, wherein over-writing is performed a configurable, plurality of times.

11. A method as claimed in claim 4, wherein the accelerator creates a meta key (K_M) and a salt (S) for access to the key server.

12. A method as claimed in claim 4, wherein the key server negotiates a session key (K_S) with the accelerator for a session, and the session key is deleted for a session.

13. A method as claimed in claim 12, wherein the key server uses the session key to encrypt data (R_C) associated with a key-creation request, and transmits the encrypted data to the accelerator.

14. A method as claimed in claim 4, wherein the management system manages a private key (K_P) of a public/private key pair as follows:

the accelerator hashes a pass phase P with a salt S to produce a per-key encryption key K_K;

the accelerator encrypts K_P with K_K;

the accelerator encrypts the result with additional data K_M, and

the accelerator returning the result to the key server.

15. A method as claimed in claim 1, wherein the key server allows access to keys only if the requesting user is already associated with a stored key.

16. A method as claimed in claim 15, wherein the management system carries out the following steps upon receiving a request from an alias for use of an existing key:

(a) the initial request is expressed in terms of P;

(b) the encrypted key is retrieved from the key store, and this is combined with P to form a request structure, R_U;

(c) R_U is encrypted with K_S and is transmitted to the accelerator;

(d) the accelerator decrypts R_U using K_S;

(e) the key is decrypted with K_M;

(f) the passphrase from the request is hashed with S to give K_K; and

(g) the result from step (e) is decrypted with K_K to give K_P, the original key.

17. A method as claimed in claim 4, wherein the key sever encrypts each key using a meta key associated with an accelerator, whereby a plurality of accelerators may use the key server.

18. A key management system comprising means for implementing a method as claimed in any preceding claim.

19. A computer program product comprising software code for performing the key server steps of claim 1 when executing on a digital computer.